# 电子科技大学

## 2018 年《数字逻辑设计与应用》

# TTL 信号频率测试计

**课程名称：**数字逻辑设计与应用

**学院名称：**光电科学与工程

**学生姓名：**李 宁　张孟迪

**学　　号：**2016050201017

2016050202014

**指导教师：**周 鹰

# 目录

# 一、课题描述

设计一 TTL 频率检测计，要求：

1、用 basys 2 板实现（时钟 50MHZ）

2、频率范围 0.1~999999.9HZ

3、用四个七段数码管分双档显示，并具备自动换挡功能，数码管尽量显示更精确的数字。

4、用 VHDL、Verilog 两种硬件描述语言实现。

5、给出正确的功能和时序仿真图。

# 二、初步构想

1、根据测频率的原理，采用在一已知时间 t0 内，记录待测信号 clock_text 上升沿的个数 N，则 f=N/t0，t0 可以用 Basys 2 开发板上自带时钟 50MHZ 分频之后产生，为了检测的方便性，我们将精度降低为 1HZ（原因见七），t0 取 1s（在 1s 内记录待测信号上升沿的个数）。

2、整个工程分为时钟产生模块 two_clk（分频 clock_led 200HZ（用于数码管显示）、clock_con 1HZ（产生控制信号模块的输入时钟））、

控制信号产生模块 contro（产生清零 clear、使能 enable（控制开始记录上升沿）、锁存 latch 信号）、

上升沿记录模块 counter_all（记录待测信号 clock_text 上升沿的个数,产生六位记录结果 led_0~5）、

锁存模块 latch（产生档位信号 D1，D2、小数点位置信号 DP_1~3、四位锁存结果 latch_0~3）、

显示模块 seg（产生显示信号 L、片选信号 N）。

3、利用另一块开发板产生测试时钟。

# 三、原理框图

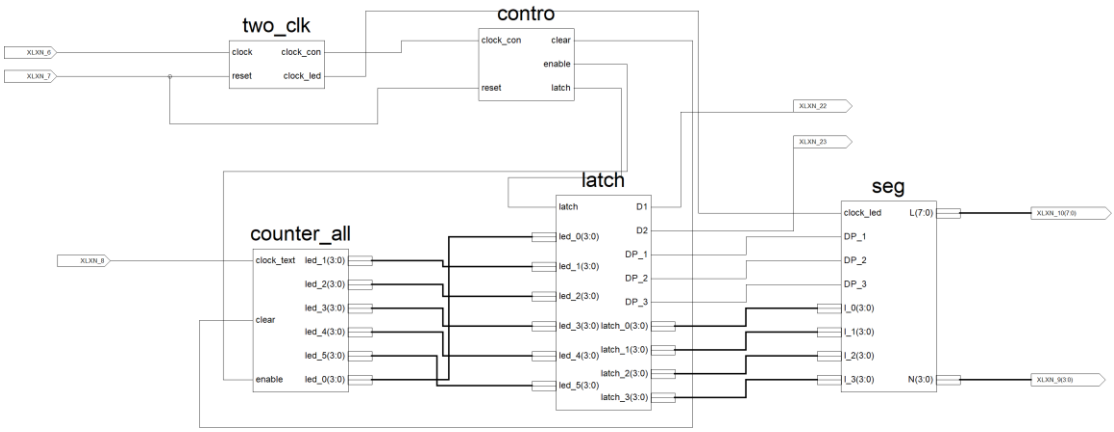


图 1

# 四、模块概述及仿真

## 1、时钟产生模块 two_clk

### 1)、简介

输入时钟 clock 为 50MHZ，分频至 200HZ（clock_led）用于数码管显示，1HZ

用于控制信号产生模块的时钟输入。

**2)、仿真波形图。**

clock

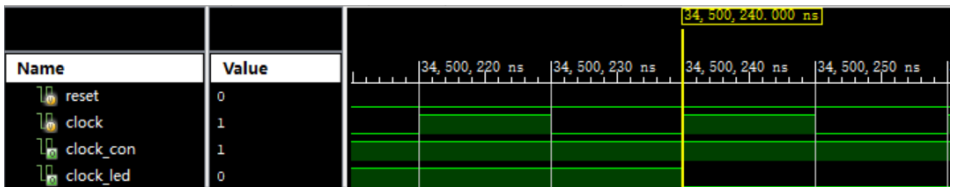


图 2

clock_con

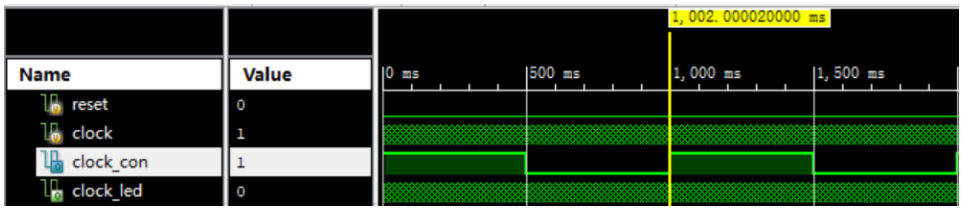


图 3

clock_led

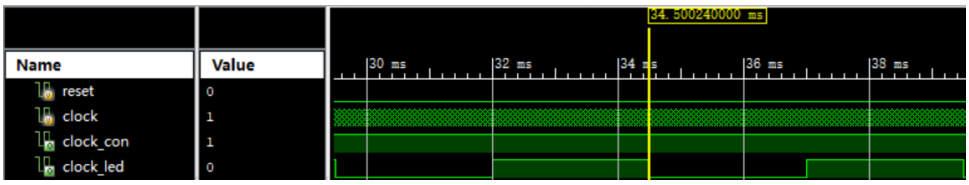


图 4

## 2、控制信号产生模块 contro

**1)、简介**

使能有效时间为整个周期 1s（上升沿记录），经过半个周期 0.5s 后，锁存信号 latch 和清零信号 clear 有效

**2)、仿真波形图**

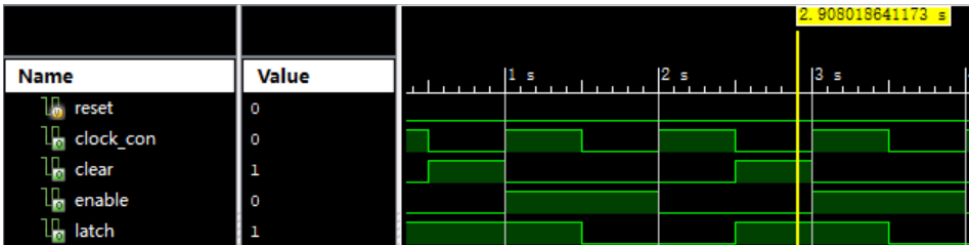


图 5

## 3、上升沿记录模块 counter_all

**1)、简介**

假想有 6 个数码管记录上升沿，从左向右依次为十万、万、千、百、十、个位，在此模块调用子模块 counter_one，实现计数满十进一的功能，并且暂时保存现有计数直到锁存，共调用 6 次，子模块分别为 U3、U4、U5、U6、U7、U8。

**2)、仿真波形图（待测信号为 1HZ）**

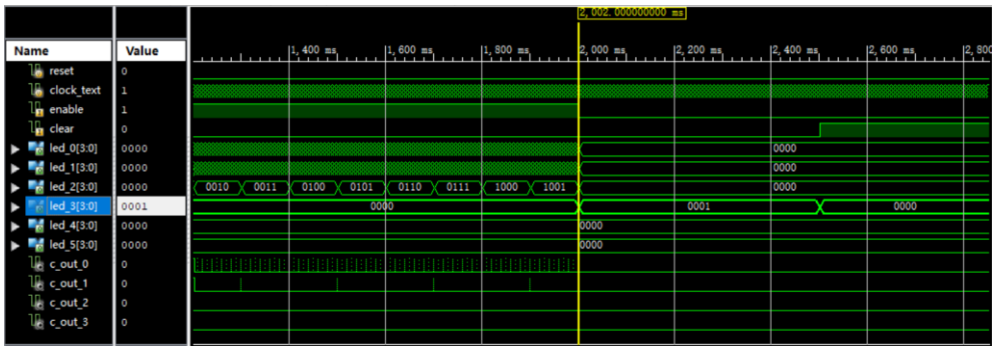


图 6

## 4、锁存模块 latch

### 1)、简介

将记录模块记录的数据锁存到 latch_0~3,并且输出档位信号 D1、D2 和小数点位置信号 DP_1~3。

### 2)、仿真波形图（待测信号为 1HZ）

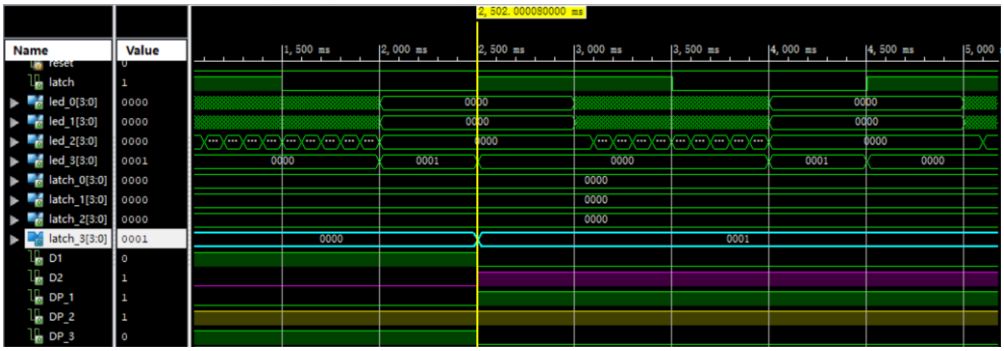


图 7

## 5、显示模块 seg

### 1)、简介

利用锁存模块产生的小数点位置信号 DP_1~3 和两个译码模块 BCD、BCD1（最低位小数点常灭）产生数码管显示信号 L 和片选信号 N。

### 2)、仿真波形图（待测信号为 1HZ）

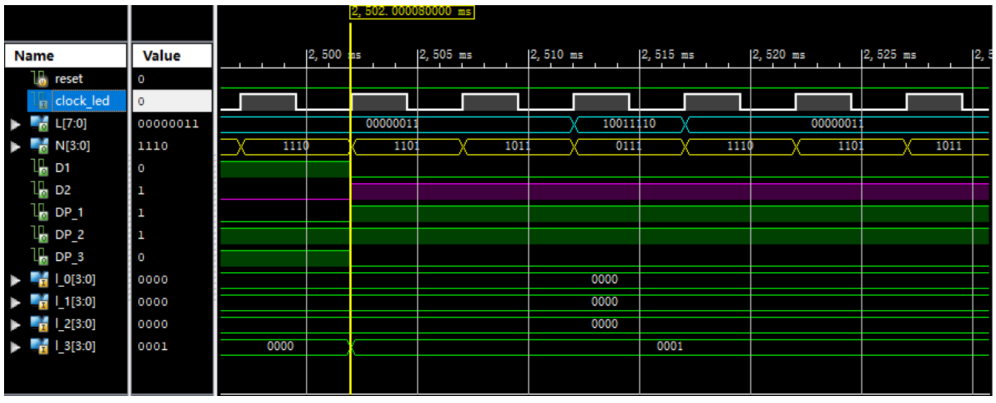


图 8

## 6、测试频率产生模块

### 1)、测试频率对应关系

| 档位 | KHZ | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 测试频率 | 961.5 | 714.2 | 500.0 | 301.2 | 100.0 | 50.00 | 10.00 | 1.000 |
| 对应倍率 | 52 | 70 | 100 | 166 | 500 | 1000 | 5000 | 50000 |
| 倍率/2 | 26 | 35 | 50 | 83 | 250 | 500 | 2500 | 25000 |
| 对应引脚 | B2 | A3 | J3 | B5 | C6 | B6 | C5 | B7 |
| 实际测试结果 | 928.1 | 696.1 | 491.3 | 298.3 | 99.83 | 50.02 | 10.02 | 1.003 |
| 档位 | HZ | | | | | | | |
| 测试频率 | 961.0 | 714.0 | 500.0 | 301.0 | 100.0 | 050.0 | 010.0 | 001.0 |
| 对应倍率 | 52000 | 70000 | 100000 | 166000 | 500000 | 1000000 | 5000000 | 50000000 |
| 倍率/2 | 26000 | 35000 | 50000 | 83000 | 250000 | 500000 | 2500000 | 25000000 |
| 对应引脚 | A9 | B9 | A10 | C9 | C12 | A13 | C13 | D12 |
| 实际测试结果 | 964.0 | 716.0 | 501.0 | 302.0 | 100.0 | 050.0 | 010.0 | 001.0 |

图 9

2）仿真波形图

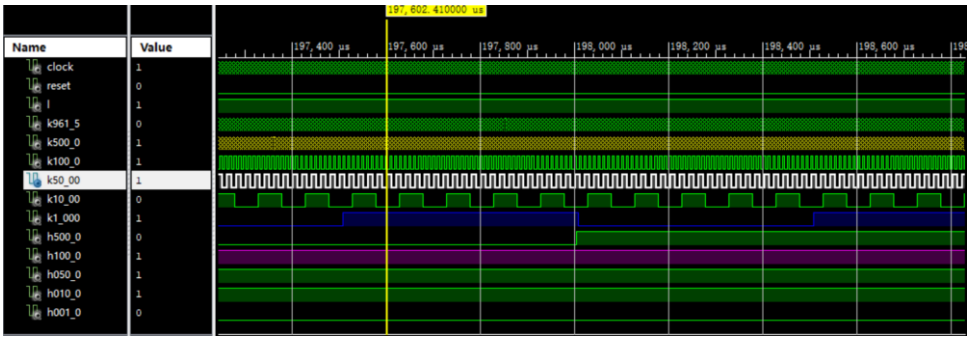


图 10

## 五、实际检测

1KHZ 如下图

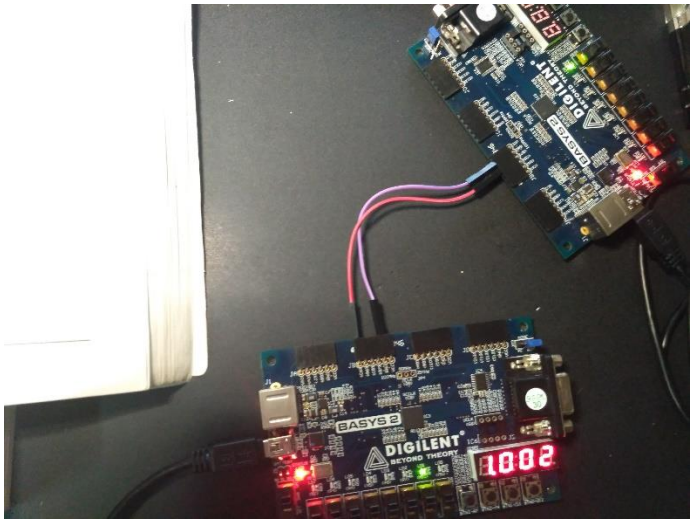


图 11

## 六、具体程序代码

详见附录

## 七、关于课程设计细节介绍

### 1、为什么将精度修改为 1HZ？

因为如果要准确测量 0.1HZ 的频率，此方案需要 10 秒钟，不利于测试，无意义。

**2、为什么加复位按键?**

由于在程序中未给一些关键信号赋初值，故从仿真的角度考虑加一个复位信号便于仿真时赋初值。

**3、为什么没有仿真改进纠错过程介绍?**

错误太多太杂，并且很多已经忘记。

**4、为何不提供时序仿真图?**

时序仿真欲观察的信号不如功能仿真易于寻找。

**5、本课程设计采用 VHDL 文件为顶层文件、基于 VHDL 原理图为顶层文件、Verilog 文件为顶层文件三种方式，两种语言。**

## 八、出现问题

**place&route 不通过，clock_text 的分配（C6 引脚）违反了约束规则**

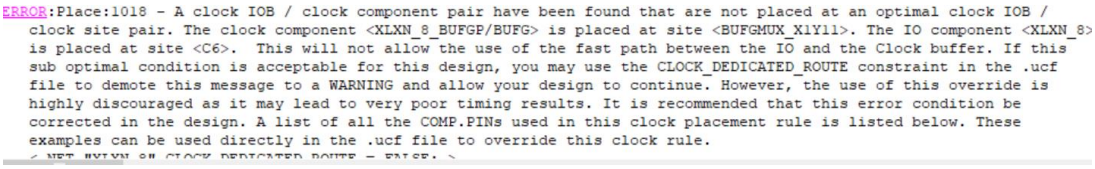


图 12

clock site pair. The clock component <XLXN_8_BUFGP/BUFG> is placed at site <BUFGMUX_X1Y11>. The IO component <XLXN_8> is placed at site <C6>. This will not allow the use of the fast path between the IO and the Clock buffer. If this sub optimal condition is acceptable for this design, you may use the CLOCK_DEDICATED_ROUTE constraint in the .ucf file to demote this message to a WARNING and allow your design to continue. However, the use of this override is highly discouraged as it may lead to very poor timing results. It is recommended that this error condition be corrected in the design. A list of all the COMP.PINs used in this clock placement rule is listed below. These examples can be used directly in the .ucf file to override this clock rule.

< NET "XLXN_8" CLOCK_DEDICATED_ROUTE = FALSE; >

**解决办法：**

**将错误提示的最后一条语句添加到 ucf 文件中即可（去掉括号）。**

## 九、关于误差的简单思考

在一定的已知时间 t0 内，上升沿测量个数最多和最少差别为 1。

## 十、总结

本次课程设计基本完成预先提出的要求，仅有一项为了测试的便利降低了精度，但这并不代表不能完成，只是完成没有任何意义。

按要求，用了两种硬件描述语言来完成课程设计，加深了对 Verilog 语言的理解，更重要的是初步了解 Verilog 与 VHDL 语言的一些区别。

## 十一、附录

**1、VHDL 代码**

**1)、HZ6**

```
----------------------------------------------------------------------
```

```
---------------
-- Company:
-- Engineer:
--
-- Create Date:    03:11:48 06/21/2018
-- Design Name:
-- Module Name:    HZ5 - Behavioral
-- Project Name:
-- Target Devices:
-- Tool versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------
---------------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--use IEEE.NUMERIC_STD.ALL;

-- Uncomment the following library declaration if instantiating
-- any Xilinx primitives in this code.
--library UNISIM;
--use UNISIM.VComponents.all;

entity HZ6 is
    Port ( clock : in  STD_LOGIC;
           reset : in std_logic;
          clock_text : in std_logic;
          D1 : out  STD_LOGIC;
          D2 : out  STD_LOGIC;
          N : out  STD_LOGIC_VECTOR (3 downto 0);
          L : out  STD_LOGIC_VECTOR (7 downto 0));
end HZ6;
```

```
architecture Behavioral of HZ6 is
-------------------------------------
signal
clock_con_t,clock_led_t,clear_t,enable_t,latch_t,D1_t,D2_t,DP_1_t,D
P_2_t,DP_3_t,
      clock_t,clock_text_t,reset_t : std_logic;
signal
led_0_t,led_1_t,led_2_t,led_3_t,led_4_t,led_5_t,led_6_t,latch_0_t,l
atch_1_t,latch_2_t,
      latch_3_t,N_t : std_logic_vector(3 downto 0);
signal L_t: std_logic_vector(7 downto 0);
-------------------------------------
component two_clk is
    Port ( clock : in  STD_LOGIC;     --50MHZ
           reset : in std_logic;
          clock_con : OUT  STD_LOGIC; --控制时钟0.1HZ
          clock_led : OUT  STD_LOGIC);--显示时钟200HZ
end component;
-------------------------------------
component contro is
    Port ( clock_con : in  STD_LOGIC;
           reset : in std_logic;
          clear : out  STD_LOGIC;
          enable : out  STD_LOGIC;
          latch : out  STD_LOGIC);
end component;
-----------------------------------------
component counter_all is
    Port ( clock_text : in  STD_LOGIC;
          clear : in  STD_LOGIC;
          enable : in  STD_LOGIC;
          led_1 : out  STD_LOGIC_VECTOR (3 downto 0);
          led_2 : out  STD_LOGIC_VECTOR (3 downto 0);
          led_3 : out  STD_LOGIC_VECTOR (3 downto 0);
          led_4 : out  STD_LOGIC_VECTOR (3 downto 0);
          led_5 : out  STD_LOGIC_VECTOR (3 downto 0);
          led_0 : out  STD_LOGIC_VECTOR (3 downto 0));
end component;
-----------------------------------------
component latch is
    Port ( latch : in  STD_LOGIC;
           D1 : out  STD_LOGIC;  --分档（LED是否亮）
            D2 : out  STD_LOGIC;
            DP_1 : OUT STD_LOGIC; --小数点的确定
```

```
            DP_2 : OUT STD_LOGIC;
            DP_3 : OUT STD_LOGIC;
          led_0 : in STD_LOGIC_VECTOR (3 downto 0);
          led_1 : in STD_LOGIC_VECTOR (3 downto 0);
          led_2 : in STD_LOGIC_VECTOR (3 downto 0);
          led_3 : in STD_LOGIC_VECTOR (3 downto 0);
          led_4 : in STD_LOGIC_VECTOR (3 downto 0);
          led_5 : in STD_LOGIC_VECTOR (3 downto 0);
          latch_0 : out  STD_LOGIC_VECTOR (3 downto 0);
          latch_1 : out  STD_LOGIC_VECTOR (3 downto 0);
          latch_2 : out  STD_LOGIC_VECTOR (3 downto 0);
          latch_3 : out  STD_LOGIC_VECTOR (3 downto 0)
            );
end component;
----------------------------------------
component seg is
    Port ( clock_led : in  STD_LOGIC;
          l_0 : in  STD_LOGIC_VECTOR (3 downto 0);
          l_1 : in  STD_LOGIC_VECTOR (3 downto 0);
          l_2 : in  STD_LOGIC_VECTOR (3 downto 0);
          l_3 : in  STD_LOGIC_VECTOR (3 downto 0);
          DP_1 : in  STD_LOGIC;
          DP_2 : in  STD_LOGIC;
          DP_3 : in  STD_LOGIC;
           L: out std_logic_vector (7 downto 0);
             N: out STD_LOGIC_VECTOR (3 downto 0)
            );
end component;
-----------------------------------------
begin
clock_t<=clock;
clock_text_t<=clock_text;
reset_t<=reset;
T1:two_clk port map (clock_t,reset_t,clock_con_t,clock_led_t);
T2:contro port map (clock_con_t,reset_t,clear_t,enable_t,latch_t);
T3:counter_all port map
(clock_text_t,clear_t,enable_t,led_1_t,led_2_t,led_3_t,led_4_t,led_
5_t,
                    led_0_t);
T4:latch port map (latch_t,D1_t,D2_t,DP_1_t,DP_2_t,DP_3_t,
              led_0_t,led_1_t,led_2_t,led_3_t,led_4_t,led_5_t,
              latch_0_t,latch_1_t,latch_2_t,latch_3_t);
T5:seg port map
(clock_led_t,latch_0_t,latch_1_t,latch_2_t,latch_3_t,DP_1_t,DP_2_t,
```

```
DP_3_t,
               L_t,N_t);
D1<=D1_t;
D2<=D2_t;
L<=L_t;
N<=N_t;

end Behavioral;
```

2)、two_clk

```
----------------------------------------------------------------------
---------------
-- Company:
-- Engineer:
--
-- Create Date:    22:13:38 06/20/2018
-- Design Name:
-- Module Name:    two_clk - Behavioral
-- Project Name:
-- Target Devices:
-- Tool versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------
---------------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;


-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--use IEEE.NUMERIC_STD.ALL;

-- Uncomment the following library declaration if instantiating
```

```
-- any Xilinx primitives in this code.
--library UNISIM;
--use UNISIM.VComponents.all;

entity two_clk is
    Port ( clock : in  STD_LOGIC;     --50MHZ
           reset : in std_logic;
          clock_con : OUT  STD_LOGIC; --控制时钟1HZ
          clock_led : OUT  STD_LOGIC);--显示时钟200HZ
end two_clk;

architecture Behavioral of two_clk is
signal s0,s1: std_logic;
signal counters: std_logic_vector (16 downto 0);
signal counter1: std_logic_vector (24 downto 0);
-----------------------------------------------
begin
process(clock,reset)
  begin
  if reset='1' then
    s0<='0';
     counters<="00000000000000000";
     else
    if (clock'event and clock ='1') then
       counters<=counters+1;
         if (counters="11110100001001000") then --片选200HZ
          s0<=not s0;
            counters<="00000000000000000";
        end if;
     end if;
  end if;
end process;
-----------------------------------
process(clock,reset)
  begin
  if reset='1' then
    s1<='0';
     counter1<="00000000000000000000000000";
     else
    if (clock'event and clock ='1') then
       counter1<=counter1+1;
         if (counter1="10111110101111000010000000") then --控制信号
1HZHZ
          s1<=not s1;
```

```
            counter1<="00000000000000000000000000";
        end if;
    end if;
  end if;
end process;
clock_led<=s0;
clock_con<=s1;
end Behavioral;
```

3)、contro

```
----------------------------------------------------------------------------------
---------------
-- Company:
-- Engineer:
--
-- Create Date:    22:51:10 06/20/2018
-- Design Name:
-- Module Name:    control - Behavioral
-- Project Name:
-- Target Devices:
-- Tool versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------------
---------------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--use IEEE.NUMERIC_STD.ALL;

-- Uncomment the following library declaration if instantiating
-- any Xilinx primitives in this code.
```

```
--library UNISIM;
--use UNISIM.VComponents.all;

entity contro is
    Port ( clock_con : in  STD_LOGIC;
           reset : in std_logic;
          clear : out  STD_LOGIC;
          enable : out  STD_LOGIC;
          latch : out  STD_LOGIC);
end contro;

architecture Behavioral of contro is
signal t1,t2:std_logic;
begin
process(clock_con,t1,t2,reset)
  begin
  if reset='1' then
    t1<='1';
     t2<='0';
     else
    if clock_con'event and clock_con='1' then
       t1<=not t1;
     end if;
     if clock_con'event and clock_con='0' then
       t2<=not t2;
       end if;
  end if;
end process;

enable<=t1;
latch<=t2;
clear<=(not clock_con)and(not t1)and(t2);

end Behavioral;
```

4)、counter_all

```
----------------------------------------------------------------------------
---------------
-- Company:
-- Engineer:
--
-- Create Date:    23:18:17 06/20/2018
```

```
-- Design Name:
-- Module Name:    counter_all - Behavioral
-- Project Name:
-- Target Devices:
-- Tool versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------
---------------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--use IEEE.NUMERIC_STD.ALL;

-- Uncomment the following library declaration if instantiating
-- any Xilinx primitives in this code.
--library UNISIM;
--use UNISIM.VComponents.all;

entity counter_all is
    Port ( clock_text : in  STD_LOGIC;
           clear : in  STD_LOGIC;
           enable : in  STD_LOGIC;
           led_1 : out  STD_LOGIC_VECTOR (3 downto 0);
           led_2 : out  STD_LOGIC_VECTOR (3 downto 0);
           led_3 : out  STD_LOGIC_VECTOR (3 downto 0);
           led_4 : out  STD_LOGIC_VECTOR (3 downto 0);
           led_5 : out  STD_LOGIC_VECTOR (3 downto 0);
           led_0 : out  STD_LOGIC_VECTOR (3 downto 0));
end counter_all;

architecture Behavioral of counter_all is
component counter_one is
Port (
```

```
      clock_con_one: in std_logic;
      clear : in  STD_LOGIC;
     c_in : in  STD_LOGIC;
     c_out : out  STD_LOGIC;
     s : out  STD_LOGIC_VECTOR (3 downto 0)
      );
end component counter_one;
signal c_out_0,c_out_1,c_out_2,c_out_3,c_out_4,c_out_5,c_out_6:
std_logic;--½øÎ»Êä³ö
begin
U3:counter_one port map(clock_text,clear,enable,c_out_0,led_0);
U4:counter_one port map(clock_text,clear,c_out_0,c_out_1,led_1);
U5:counter_one port map(clock_text,clear,c_out_1,c_out_2,led_2);
U6:counter_one port map(clock_text,clear,c_out_2,c_out_3,led_3);
U7:counter_one port map(clock_text,clear,c_out_3,c_out_4,led_4);
U8:counter_one port map(clock_text,clear,c_out_4,c_out_5,led_5);

end Behavioral;
```

5)、latch

```
----------------------------------------------------------------------
---------------
-- Company:
-- Engineer:
--
-- Create Date:    23:30:55 06/20/2018
-- Design Name:
-- Module Name:    latch - Behavioral
-- Project Name:
-- Target Devices:
-- Tool versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------
---------------
library IEEE;
```

```
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--use IEEE.NUMERIC_STD.ALL;

-- Uncomment the following library declaration if instantiating
-- any Xilinx primitives in this code.
--library UNISIM;
--use UNISIM.VComponents.all;

entity latch is
    Port ( latch : in  STD_LOGIC;
           D1 : out  STD_LOGIC;  --分档（LED是否亮）
             D2 : out  STD_LOGIC;
             DP_1 : OUT STD_LOGIC; --小数点的确定
             DP_2 : OUT STD_LOGIC;
             DP_3 : OUT STD_LOGIC;
          led_0 : in STD_LOGIC_VECTOR (3 downto 0);
          led_1 : in STD_LOGIC_VECTOR (3 downto 0);
          led_2 : in STD_LOGIC_VECTOR (3 downto 0);
          led_3 : in STD_LOGIC_VECTOR (3 downto 0);
          led_4 : in STD_LOGIC_VECTOR (3 downto 0);
          led_5 : in STD_LOGIC_VECTOR (3 downto 0);
          latch_0 : out  STD_LOGIC_VECTOR (3 downto 0);
          latch_1 : out  STD_LOGIC_VECTOR (3 downto 0);
          latch_2 : out  STD_LOGIC_VECTOR (3 downto 0);
          latch_3 : out  STD_LOGIC_VECTOR (3 downto 0)
            );
end latch;

architecture Behavioral of latch is

begin
process(latch)
  variable latch_led_0 : STD_LOGIC_VECTOR (3 downto 0);
  variable latch_led_1 : STD_LOGIC_VECTOR (3 downto 0);
  variable latch_led_2 : STD_LOGIC_VECTOR (3 downto 0);
  variable latch_led_3 : STD_LOGIC_VECTOR (3 downto 0);
  variable latch_led_4 : STD_LOGIC_VECTOR (3 downto 0);
  variable latch_led_5 : STD_LOGIC_VECTOR (3 downto 0);
  begin
```

```
    if latch'event and latch='1' then
        latch_led_0:=led_0;
         latch_led_1:=led_1;
         latch_led_2:=led_2;
         latch_led_3:=led_3;
         latch_led_4:=led_4;
         latch_led_5:=led_5;
         if latch_led_5="0000" then
            if latch_led_4="0000" then
               if latch_led_3="0000" then
                  D1<='1';
                  D2<='0';
                  latch_0<="0000";
                  latch_1<=latch_led_0;
                  latch_2<=latch_led_1;
                  latch_3<=latch_led_2;
              DP_1<='0';
              DP_2<='1';
              DP_3<='1';
             else
                  D1<='0';
                  D2<='1';
                  latch_0<=latch_led_0;
                  latch_1<=latch_led_1;
                  latch_2<=latch_led_2;
                  latch_3<=latch_led_3;
              DP_1<='1';
              DP_2<='1';
              DP_3<='0';
            end if;
          else
               D1<='0';
               D2<='1';
               latch_0<=latch_led_1;
               latch_1<=latch_led_2;
               latch_2<=latch_led_3;
               latch_3<=latch_led_4;
            DP_1<='1';
            DP_2<='0';
            DP_3<='1';
         end if;
         else
            D1<='0';
            D2<='1';
```

```
            latch_0<=latch_led_2;
            latch_1<=latch_led_3;
            latch_2<=latch_led_4;
            latch_3<=latch_led_5;
          DP_1<='0';
          DP_2<='1';
          DP_3<='1';
         end if;
        end if;
end process;
end Behavioral;
```

6)、seg

```
----------------------------------------------------------------------
---------------
-- Company:
-- Engineer:
--
-- Create Date:    00:08:33 06/21/2018
-- Design Name:
-- Module Name:    seg - Behavioral
-- Project Name:
-- Target Devices:
-- Tool versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------
---------------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--use IEEE.NUMERIC_STD.ALL;
```

```
-- Uncomment the following library declaration if instantiating
-- any Xilinx primitives in this code.
--library UNISIM;
--use UNISIM.VComponents.all;

entity seg is
    Port ( clock_led : in  STD_LOGIC;
           l_0 : in  STD_LOGIC_VECTOR (3 downto 0);
           l_1 : in  STD_LOGIC_VECTOR (3 downto 0);
           l_2 : in  STD_LOGIC_VECTOR (3 downto 0);
           l_3 : in  STD_LOGIC_VECTOR (3 downto 0);
           DP_1 : in  STD_LOGIC;
           DP_2 : in  STD_LOGIC;
           DP_3 : in  STD_LOGIC;
            L: out std_logic_vector (7 downto 0);
              N: out STD_LOGIC_VECTOR (3 downto 0)
             );
end seg;

architecture Behavioral of seg is
type state_type is (D1,C1,B1,A1);
signal sreg,snext: state_type;
signal A,B,C,D: std_logic_vector (7 downto 0);
signal s0,s1,s2,s3 : std_logic_vector (3 downto 0);
-------------------------------------------------
component BCD port (
      DP: in std_logic;
      A: out STD_LOGIC_VECTOR (7 downto 0);
        S: in STD_LOGIC_VECTOR (3 downto 0)
        );
end component;
---------------------------------------
component BCD1 port (
      A: out STD_LOGIC_VECTOR (7 downto 0);
        S: in STD_LOGIC_VECTOR (3 downto 0)
        );
end component;
-----------------------------------------------
begin
process(clock_led)
begin
  if clock_led'event and clock_led='1' then
    sreg <=snext;
```

```
  end if;
end process;

process(sreg)  --鍵盤掃描 200HZ
  begin
       case sreg is
           when A1    =>  snext <=B1;
           when B1    =>  snext <=C1;
           when C1    =>  snext <=D1;
           when D1    =>  snext <=A1;
           when others => snext <=A1;
       end case;
end process;
  s0<=l_0;
  s1<=l_1;
  s2<=l_2;
  s3<=l_3;
  U10:BCD1 port map (A,s0);
  U11:BCD port map (DP_1,B,s1);
  U12:BCD port map (DP_2,C,s2);
  U13:BCD port map (DP_3,D,s3);

  with sreg select
    L <= A when A1,
         B when B1,
         C when C1,
         D when D1;

  with sreg select
    N <=  "1110" when A1,
          "1101" when B1,
          "1011" when C1,
          "0111" when D1;

end Behavioral;
```

7)、counter_one

```
----------------------------------------------------------------------------
---------------
-- Company:
-- Engineer:
--
```

```
-- Create Date:    23:02:26 06/20/2018
-- Design Name:
-- Module Name:    counter_one - Behavioral
-- Project Name:
-- Target Devices:
-- Tool versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------
---------------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
USE IEEE.STD_LOGIC_UNSIGNED.ALL;

-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--use IEEE.NUMERIC_STD.ALL;

-- Uncomment the following library declaration if instantiating
-- any Xilinx primitives in this code.
--library UNISIM;
--use UNISIM.VComponents.all;

entity counter_one is
    Port (
           clock_con_one: in std_logic;
           clear : in  STD_LOGIC;
          c_in : in  STD_LOGIC;
          c_out : out  STD_LOGIC;
          s : out  STD_LOGIC_VECTOR (3 downto 0)
            );
end counter_one;

architecture Behavioral of counter_one is
signal counter: std_logic_vector(3 downto 0);
begin
process(clear,clock_con_one)
```

```
begin
  if clear='1' then
    counter<="0000";
  else if clock_con_one'event and clock_con_one='1' then
       if c_in='1' then
           if counter<"1001" then
              counter<=counter+1;
            else
              counter<="0000";
            end if;
         else
         null;
         end if;
      end if;
  end if;
end process;
s<=counter;
c_out<='1' when c_in='1' and counter="1001" else '0';
end Behavioral;
```

8)、BCD

```
----------------------------------------------------------------------------
---------------
-- Company:
-- Engineer:
--
-- Create Date:    20:53:07 05/10/2018
-- Design Name:
-- Module Name:    BCD - Behavioral
-- Project Name:
-- Target Devices:
-- Tool versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------
---------------
```

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--use IEEE.NUMERIC_STD.ALL;

-- Uncomment the following library declaration if instantiating
-- any Xilinx primitives in this code.
--library UNISIM;
--use UNISIM.VComponents.all;

entity BCD is
port (
     DP: in std_logic;
     A: out STD_LOGIC_VECTOR (7 downto 0);
       S: in STD_LOGIC_VECTOR (3 downto 0)
       );
end BCD;

architecture Behavioral of BCD is
begin

process(DP,S)
 begin
   if DP='1' then
     case S is        --abcdefg DP
          when "0000" => A <= "00000011";
           when "0001" => A <= "10011111";
           when "0010" => A <= "00100101";
           when "0011" => A <= "00001101";
           when "0100" => A <= "10011001";
           when "0101" => A <= "01001001";
           when "0110" => A <= "01000001";
           when "0111" => A <= "00011111";
           when "1000" => A <= "00000001";
           when "1001" => A <= "00001001";
           when "1010" => A <= "11111101";
           when others => A <= "00000011";
      end case;
    else
     case S is        --abcdefg DP
          when "0000" => A <= "00000010";
           when "0001" => A <= "10011110";
```

```
          when "0010" => A <= "00100100";
          when "0011" => A <= "00001100";
          when "0100" => A <= "10011000";
          when "0101" => A <= "01001000";
          when "0110" => A <= "01000000";
          when "0111" => A <= "00011110";
          when "1000" => A <= "00000000";
          when "1001" => A <= "00001000";
          when "1010" => A <= "11111100";
          when others => A <= "00000010";
      end case;
     end if;
end process;
end Behavioral;
```

9)、BCD1

```
----------------------------------------------------------------------------
---------------
-- Company:
-- Engineer:
--
-- Create Date:    20:53:07 05/10/2018
-- Design Name:
-- Module Name:    BCD - Behavioral
-- Project Name:
-- Target Devices:
-- Tool versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------
---------------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
```

```
--use IEEE.NUMERIC_STD.ALL;

-- Uncomment the following library declaration if instantiating
-- any Xilinx primitives in this code.
--library UNISIM;
--use UNISIM.VComponents.all;

entity BCD1 is
port (
     A: out STD_LOGIC_VECTOR (7 downto 0);
       S: in STD_LOGIC_VECTOR (3 downto 0)
       );
end BCD1;

architecture Behavioral of BCD1 is
begin

process(S)
  begin
    case S is        --abcdefg
        when "0000" => A <= "00000011";
         when "0001" => A <= "10011111";
         when "0010" => A <= "00100101";
         when "0011" => A <= "00001101";
         when "0100" => A <= "10011001";
         when "0101" => A <= "01001001";
         when "0110" => A <= "01000001";
         when "0111" => A <= "00011111";
         when "1000" => A <= "00000001";
         when "1001" => A <= "00001001";
         when "1010" => A <= "11111101";
         when others => A <= "00000011";
         end case;
end process;
end Behavioral;
```

10)、text

```
----------------------------------------------------------------------------
-------------
-- Company:
-- Engineer:
--
```

```
-- Create Date:   00:37:31 06/22/2018
-- Design Name:
-- Module Name:   D:/ISE_project/HZ6/text.vhd
-- Project Name:  HZ6
-- Target Device:
-- Tool versions:
-- Description:
--
-- VHDL Test Bench Created by ISE for module: HZ6
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
-- Notes:
-- This testbench has been automatically generated using types
std_logic and
-- std_logic_vector for the ports of the unit under test.  Xilinx
recommends
-- that these types always be used for the top-level I/O of a
design in order
-- to guarantee that the testbench will bind correctly to the post-
implementation
-- simulation model.
--------------------------------------------------------------------
-------------
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;

-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--USE ieee.numeric_std.ALL;

ENTITY text IS
END text;

ARCHITECTURE behavior OF text IS

    -- Component Declaration for the Unit Under Test (UUT)

    COMPONENT HZ6
    PORT(
```

```
        clock : IN  std_logic;
        reset : IN  std_logic;
        clock_text : IN  std_logic;
        D1 : OUT  std_logic;
        D2 : OUT  std_logic;
        N : OUT  std_logic_vector(3 downto 0);
        L : OUT  std_logic_vector(7 downto 0)
       );
    END COMPONENT;


   --Inputs
   signal clock : std_logic := '0';
   signal reset : std_logic := '0';
   signal clock_text : std_logic := '0';

    --Outputs
   signal D1 : std_logic;
   signal D2 : std_logic;
   signal N : std_logic_vector(3 downto 0);
   signal L : std_logic_vector(7 downto 0);

   -- Clock period definitions
   constant clock_period : time := 10 ns;
   constant clock_text_period : time := 10 ns;

BEGIN

    -- Instantiate the Unit Under Test (UUT)
   uut: HZ6 PORT MAP (
        clock => clock,
        reset => reset,
        clock_text => clock_text,
        D1 => D1,
        D2 => D2,
        N => N,
        L => L
       );

   -- Clock process definitions
   clock_process :process
   begin
        clock <= '0';
        wait for 10 ns;
```

```
        clock <= '1';
        wait for 10 ns;
    end process;

    clock_text_process :process
    begin
        clock_text <= '0';
        wait for 500 us;
        clock_text <= '1';
        wait for 500 us;
    end process;


    -- Stimulus process
    stim_proc: process
      begin
          reset<='1';
          wait for 1 ms;
          reset<='0';
          wait for 10000 ms;
    end process;

END;
```

2、Verilog 代码

1)、HZV

```
`timescale 1ns / 1ps
//////////////////////////////////////////////////////////////////////////////////
// Company:
// Engineer:
//
// Create Date:    15:20:28 07/04/2018
// Design Name:
// Module Name:    HZV
// Project Name:
// Target Devices:
// Tool versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
```

```
// Additional Comments:
//
//////////////////////////////////////////////////////////////////////////////////
///////////////
module HZV(
    input clock,
    input reset,
    input clock_text,
    output D1,
    output D2,
    output [3:0] N,
    output [7:0] L
    );
//////////////////////////////////////
wire
clock_con_t,clock_led_t,clear_t,enable_t,latch_t,D1_t,D2_t,DP_1_t,D
P_2_t,DP_3_t,
      clock_t,clock_text_t,reset_t ;
wire [3:0]
led_0_t,led_1_t,led_2_t,led_3_t,led_4_t,led_5_t,led_6_t,latch_0_t,l
atch_1_t,latch_2_t,
      latch_3_t,N_t ;
wire [7:0] L_t ;
//////////////////////////////////////
assign clock_t=clock;
assign clock_text_t=clock_text;
assign reset_t=reset;
two_clk T1 (clock_t,reset_t,clock_con_t,clock_led_t);
contro T2 (clock_con_t,reset_t,clear_t,enable_t,latch_t);
counter_all T3
(clock_text_t,clear_t,enable_t,led_1_t,led_2_t,led_3_t,led_4_t,led_
5_t,
                    led_0_t);
latch T4 (latch_t,D1_t,D2_t,DP_1_t,DP_2_t,DP_3_t,
              led_0_t,led_1_t,led_2_t,led_3_t,led_4_t,led_5_t,
              latch_0_t,latch_1_t,latch_2_t,latch_3_t);
seg T5
(clock_led_t,latch_0_t,latch_1_t,latch_2_t,latch_3_t,DP_1_t,DP_2_t,
DP_3_t,
              L_t,N_t);
//////////////////////////////////
assign D1=D1_t;
assign D2=D2_t;
assign L=L_t;
```

```
assign N=N_t;

endmodule
```

2)、two_clk

```
`timescale 1ns / 1ps
//////////////////////////////////////////////////////////////////////////////////
// Company:
// Engineer:
//
// Create Date:    16:17:39 07/04/2018
// Design Name:
// Module Name:    two_clk
// Project Name:
// Target Devices:
// Tool versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
//////////////////////////////////////////////////////////////////////////////////
module two_clk(
    input clock,
    input reset,
    output clock_con,
    output clock_led
    );
reg s0,s1;
reg [16:0] counters;
reg [24:0] counter1;

always@(posedge clock or posedge reset)
  begin
    if (reset)
      begin
         s0<=1;
         counters<=17'b0_0000_0000_0000_0000;
```

```
        end
    else
        begin
          counters<=counters+1;
            if (counters==17'b1_1110_1000_0100_1000) //--Tw200HZ
                begin
                   s0<=!s0;
                   counters<=17'b0_0000_0000_0000_0000;
                end
        end
  end
////////////////////////////////////////////////
always@(posedge clock or posedge reset)
  begin
    if (reset)
      begin
         s1<=1;
         counter1<=25'b0_0000_0000_0000_0000_0000_0000;
       end
    else
      begin
         counter1<=counter1+1;
          if (counter1==25'b1_0111_1101_0111_1000_0100_0000) //--
ʊ$S1¿HZ
             begin
               s1<=!s1;
               counter1<=25'b0_0000_0000_0000_0000_0000_0000;
             end
       end
  end
//////////////////////////
assign clock_led=s0;
assign clock_con=s1;
endmodule
```

3)、contro

```
`timescale 1ns / 1ps
//////////////////////////////////////////////////////////////////////////////////
////////////////
// Company:
// Engineer:
//
// Create Date:    17:42:05 07/04/2018
```

```
// Design Name:
// Module Name:    contro
// Project Name:
// Target Devices:
// Tool versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
//////////////////////////////////////////////////////////////////////////////////
module contro(
    input clock_con,
    input reset,
    output clear,
    output enable,
    output latch
    );
reg t1,t2;

always@(posedge clock_con or posedge reset)
  begin
  if (reset)
     t1<=0;
  else t1<=!t1;
  end

always@(negedge clock_con or posedge reset)
  begin
  if (reset)
     t2<=1;
  else t2<=!t1;
  end
assign enable=t1;
assign latch=t2;
assign clear=(!clock_con)&&(!t1)&&(t2);
endmodule
```

4)、counter_all

```
`timescale 1ns / 1ps
//////////////////////////////////////////////////////////////////////////////////
// Company:
// Engineer:
//
// Create Date:    18:43:58 07/04/2018
// Design Name:
// Module Name:    counter_all
// Project Name:
// Target Devices:
// Tool versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
//////////////////////////////////////////////////////////////////////////////////
module counter_all(
    input clock_text,
    input clear,
    input enable,
    output [3:0] led_1,
    output [3:0] led_2,
    output [3:0] led_3,
    output [3:0] led_4,
    output [3:0] led_5,
    output [3:0] led_0
    );
////////////////////////////////
wire c_out_0,c_out_1,c_out_2,c_out_3,c_out_4,c_out_5,c_out_6;
////////////////////////////////
counter_one U3 (clock_text,clear,enable,c_out_0,led_0);
counter_one U4 (clock_text,clear,c_out_0,c_out_1,led_1);
counter_one U5 (clock_text,clear,c_out_1,c_out_2,led_2);
counter_one U6 (clock_text,clear,c_out_2,c_out_3,led_3);
counter_one U7 (clock_text,clear,c_out_3,c_out_4,led_4);
counter_one U8 (clock_text,clear,c_out_4,c_out_5,led_5);

endmodule
```

5)、latch

```
`timescale 1ns / 1ps
//////////////////////////////////////////////////////////////////////////////////
// Company:
// Engineer:
//
// Create Date:    18:52:35 07/04/2018
// Design Name:
// Module Name:    latch
// Project Name:
// Target Devices:
// Tool versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
//////////////////////////////////////////////////////////////////////////////////
module latch(
    input latch,
    output D1,
    output D2,
    output DP_1,
    output DP_2,
    output DP_3,
    input [3:0] led_0,
    input [3:0] led_1,
    input [3:0] led_2,
    input [3:0] led_3,
    input [3:0] led_4,
    input [3:0] led_5,
    output [3:0] latch_0,
    output [3:0] latch_1,
    output [3:0] latch_2,
    output [3:0] latch_3
    );
//////////////////////////////
```

```
reg latch_D1,latch_D2,latch_DP_1,latch_DP_2,latch_DP_3;
reg [3:0] latch_latch_0,latch_latch_1,latch_latch_2,latch_latch_3;
////////////////////////////////
always@(posedge latch)
  begin
  if (led_5==4'b0000)
     if (led_4==4'b0000)
         if (led_3==4'b0000)
             begin
                latch_D1<=1;
                latch_D2<=0;
                latch_latch_0<=4'b0000;
                latch_latch_1<=led_0;
                latch_latch_2<=led_1;
                latch_latch_3<=led_2;
                 latch_DP_1<=0;
             latch_DP_2<=1;
             latch_DP_3<=1;
              end
          else
             begin
                  latch_D1<=0;
                  latch_D2<=1;
                  latch_latch_0<=led_0;
                  latch_latch_1<=led_1;
                  latch_latch_2<=led_2;
                  latch_latch_3<=led_3;
              latch_DP_1<=1;
              latch_DP_2<=1;
              latch_DP_3<=0;
               end
      else
        begin
           latch_D1<=0;
               latch_D2<=1;
               latch_latch_0<=led_1;
               latch_latch_1<=led_2;
               latch_latch_2<=led_3;
               latch_latch_3<=led_4;
           latch_DP_1<=1;
           latch_DP_2<=0;
           latch_DP_3<=1;
          end
  else
```

```
    begin
        latch_D1<=0;
            latch_D2<=1;
            latch_latch_0<=led_2;
            latch_latch_1<=led_3;
            latch_latch_2<=led_4;
            latch_latch_3<=led_5;
         latch_DP_1<=0;
         latch_DP_2<=1;
         latch_DP_3<=1;
     end
  end
//////////////////////////
assign D1=latch_D1;
assign D2=latch_D2;
assign DP_1=latch_DP_1;
assign DP_2=latch_DP_2;
assign DP_3=latch_DP_3;
assign latch_0=latch_latch_0;
assign latch_1=latch_latch_1;
assign latch_2=latch_latch_2;
assign latch_3=latch_latch_3;
endmodule
```

6)、seg

```
`timescale 1ns / 1ps
//////////////////////////////////////////////////////////////////////////////////
////////////////
// Company:
// Engineer:
//
// Create Date:    19:41:12 07/04/2018
// Design Name:
// Module Name:    seg
// Project Name:
// Target Devices:
// Tool versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
```

```
// Additional Comments:
//
//////////////////////////////////////////////////////////////////////////////////
///////////////
module seg(
    input clock_led,
    input [3:0] l_0,
    input [3:0] l_1,
    input [3:0] l_2,
    input [3:0] l_3,
    input DP_1,
    input DP_2,
    input DP_3,
    output [7:0] L,
    output [3:0] N
    );
////////////////////////////////////
reg [1:0] sreg=2'b00,
          snext=2'b01;
parameter [1:0] A1=2'b00,
                B1=2'b01,
                     C1=2'b10,
                     D1=2'b11;
reg [7:0] L0;
wire [7:0] A,B,C,D;
reg [4:0] N0;
////////////////////////////////
always@(posedge clock_led)
  sreg<=snext;
////////////////////////////////
always@(sreg)
  begin
    case (sreg)
        A1: snext=B1;
        B1: snext=C1;
        C1: snext=D1;
        D1: snext=A1;
        default snext=A1;
      endcase
  end
////////////////////////////////
BCD1 U10 (A,l_0);
BCD U11 (DP_1,B,l_1);
BCD U12 (DP_2,C,l_2);
```

```
BCD U13 (DP_3,D,l_3);
//////////////////////////////
always@(sreg,A,B,C,D)
 begin
    case (sreg)
        A1: L0<=A;
        B1: L0<=B;
        C1: L0<=C;
        D1: L0<=D;
        default L0<=A;
     endcase
 end
//////////////////////
always@(sreg)
 begin
    case (sreg)
        A1: N0<=4'b1110;
        B1: N0<=4'b1101;
        C1: N0<=4'b1011;
        D1: N0<=4'b0111;
        default N0<=4'b1110;
     endcase
 end
//////////////////////////
assign L=L0;
assign N=N0;
endmodule
```

7)、counter_one

```
`timescale 1ns / 1ps
//////////////////////////////////////////////////////////////////////////////////
////////////////
// Company:
// Engineer:
//
// Create Date:    18:00:48 07/04/2018
// Design Name:
// Module Name:    counter_one
// Project Name:
// Target Devices:
// Tool versions:
// Description:
//
```

```
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
//////////////////////////////////////////////////////////////////////////////////
////////////////
module counter_one(
    input clock_con_one,
    input clear,
    input c_in,
    output c_out,
    output [3:0] s
    );
//////////////////////
reg [3:0] counter=4'd0000;
//////////////////////
always@(posedge clock_con_one)
  begin
  if (clear)
     begin
       counter<=4'b0000;
      end
  else
    begin
       if (c_in)
           if (counter<4'b1001)
               begin
                 counter<=counter+1;
                end
            else
               begin
                 counter<=4'b0000;
                end
     end
  end
/////////////////
assign s=counter;
assign c_out=(counter==4'b1001)&&c_in;
endmodule
```

8)、BCD

```
`timescale 1ns / 1ps
//////////////////////////////////////////////////////////////////////////////////
// Company:
// Engineer:
//
// Create Date:    19:18:36 07/04/2018
// Design Name:
// Module Name:    BCD
// Project Name:
// Target Devices:
// Tool versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
//////////////////////////////////////////////////////////////////////////////////
module BCD(
    input DP,
    output [7:0] A,
    input [3:0] S
    );
////////////////////////////
reg [7:0] BA;
////////////////////////////
always@(DP,S)
  begin
    if (DP==1'b1)
        begin
           case (S)      //abcdefg DP
              4'b0000: BA=8'b00000011;
              4'b0001: BA=8'b10011111;
              4'b0010: BA=8'b00100101;
              4'b0011: BA=8'b00001101;
              4'b0100: BA=8'b10011001;
              4'b0101: BA=8'b01001001;
              4'b0110: BA=8'b01000001;
              4'b0111: BA=8'b00011111;
              4'b1000: BA=8'b00000001;
```

```
            4'b1001: BA=8'b00001001;
            4'b1010: BA=8'b11111101;
            default BA=8'b00000011;
          endcase
        end
    else
      begin
        case (S)      //abcdefg DP
            4'b0000: BA=8'b00000010;
            4'b0001: BA=8'b10011110;
            4'b0010: BA=8'b00100100;
            4'b0011: BA=8'b00001100;
            4'b0100: BA=8'b10011000;
            4'b0101: BA=8'b01001000;
            4'b0110: BA=8'b01000000;
            4'b0111: BA=8'b00011110;
            4'b1000: BA=8'b00000000;
            4'b1001: BA=8'b00001000;
            4'b1010: BA=8'b11111100;
            default BA=8'b00000010;
          endcase
        end
  end
//////////////////////////
assign A=BA;

endmodule
```

9)、BCD1

```
`timescale 1ns / 1ps
//////////////////////////////////////////////////////////////////////////////////
////////////////
// Company:
// Engineer:
//
// Create Date:    19:34:27 07/04/2018
// Design Name:
// Module Name:    BCD1
// Project Name:
// Target Devices:
// Tool versions:
// Description:
//
```

```
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
//////////////////////////////////////////////////////////////////////////////////
////////////////
module BCD1(
    output [7:0] A,
    input [3:0] S
    );
//////////////////////////////
reg [7:0] BA;
//////////////////////////////
always@(S)
  begin
     case (S)      //abcdefg DP
        4'b0000: BA=8'b00000011;
        4'b0001: BA=8'b10011111;
        4'b0010: BA=8'b00100101;
        4'b0011: BA=8'b00001101;
        4'b0100: BA=8'b10011001;
        4'b0101: BA=8'b01001001;
        4'b0110: BA=8'b01000001;
        4'b0111: BA=8'b00011111;
        4'b1000: BA=8'b00000001;
        4'b1001: BA=8'b00001001;
        4'b1010: BA=8'b11111101;
        default BA=8'b00000011;
      endcase
  end
//////////////////////////////
assign A=BA;
endmodule
```

10)、text

```
`timescale 1ns / 1ps

//////////////////////////////////////////////////////////////////////////////////
//////////////
// Company:
// Engineer:
```

```
//
// Create Date:   20:34:43 07/04/2018
// Design Name:   HZV
// Module Name:   D:/ISE_project/HZV/text.v
// Project Name:  HZV
// Target Device:
// Tool versions:
// Description:
//
// Verilog Test Fixture created by ISE for module: HZV
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
////////////////////////////////////////////////////////////////////////
/////////////

module text;

    // Inputs
    reg clock;
    reg reset;
    reg clock_text;

    // Outputs
    wire D1;
    wire D2;
    wire [3:0] N;
    wire [7:0] L;

    // Instantiate the Unit Under Test (UUT)
    HZV uut (
        .clock(clock),
        .reset(reset),
        .clock_text(clock_text),
        .D1(D1),
        .D2(D2),
        .N(N),
        .L(L)
    );
```

```
    initial begin
        // Initialize Inputs
        reset = 0;

        // Wait 100 ns for global reset to finish
        #1000000 reset=1;
        #1000000 reset=0;
    end
    always
      begin
        #10 clock=0;
        #10 clock=1;
      end
   always
      begin
        #500000 clock_text=0;
        #500000 clock_text=1;
      end
endmodule
```

## 十二、参考资料

【1】数字设计原理与实践（原书第四版） 林生 葛红 金京林 译 机械工业出版社 2016 年 3 月第 7 次印刷。

【2】FPGA/CPLD 设计与实践教程 沈莉丽 卢家凰 张志立 编 中国电力出版社出版 2017 年 1 月第一版。

【3】Verilog 经典教程 夏雨闻 著 北京航空航天大学出版社

【4】VHDL 数字电路设计教程 乔庐峰 王志功 等译 电子工业出版社 2009 年 12 月第 5 次印刷